IN THE CLAIMS

Please amend the claims as follows:

Claim 1 (Original): A method for using a shared library called up from a calling source program in a tamper resistant microprocessor which has a function for decrypting and executing encrypted codes and a table formed by a plurality of regions for storing a plurality of encryption keys corresponding to at least one program and at least one shared library to be called up by the at least one program, the method comprising:

creating a task for the shared library;

allocating a task identifier to the task;

acquiring an instruction key from a header of the shared library;

storing the instruction key into a region of the table corresponding to the task identifier allocated to the task for the shared library in the microprocessor;

initializing by executing a loader in the shared library; and

returning a control to the calling source program via an entry point in the shared library.

Claim 2 (Original): The method of claim 1, further comprising:

loading another shared library by referring to an import table in the shared library, after storing the instruction key into the table in the microprocessor.

Claim 3 (Currently Amended): The method of claim 1, wherein the initializing step executes as many loaders in the shared library as a number of calling source programs, and the method further comprising:

producing as many data keys as the number of calling source programs for encrypting data to be used by the shared library, before the returning step;

storing the data keys into a region of the table to which the task identifier of the task for the shared library is allocated in the microprocessor, before the returning step;

setting the shared library in a standby state waiting for a call up from the calling source program, after the returning step;

having the shared library authenticated by the calling source program;

receiving an address of a shared memory region produced by the calling source program;

setting the shared memory region as a shared encrypted data region to be used in data exchange between the calling source program and the shared library;

controlling the shared ~~libraryt~~ library to receive a signal for calling up a sub-routine in the shared library from the calling source program;

verifying a checksum of data of the calling source program;

carrying out a processing requested from the calling source program when the checksum of the data of the calling source program matches the data; and

sending a result of the requested processing by adding the checksum into the shared encrypted data region.

Claim 4 (Original): The method of claim 3, further comprising:

encrypting a work memory region for carrying out the processing requested from the calling source program by using one of the data keys of the shared library, after the verifying step;

wherein the carrying out step carries out the processing requested from the calling source program in the work memory region encrypted by the encrypting step.

Claim 5 (Original): The method of claim 3, wherein the authenticating step carries out an authentication by sending to the calling source program an authentication information to which a signature according to a secret key stored in the shared library in advance is attached, and producing a common key to be used between the calling source program and the shared library.

Claim 6 (Original): The method of claim 5, further comprising:

encrypting the common key by using one of the data keys of the shared library, after the authenticating step; and

storing the common key encrypted by the encrypting step into a secret data region of the shared library.

Claim 7 (Original): The method of claim 1, further comprising:

generating a data key for encrypting a work memory region of the shared library from a random number from a random number generation unit in the microprocessor before the returning step; and

setting the shared library in a standby state waiting for a call up from the calling source program, after the returning step.

Claims 8-9 (Canceled).